Real-time 3D Human Pose and Motion Reconstruction from Monocular RGB Videos

Anastasion Yiannakides Andreas Aristidou

Yiorgos Chrysanthou

University of Cyprus

75, Kallipoleos Street

1678 Nicosia, Cyprus

email: tasyiann@gmail.com, a.aristidou@ieee.org, yiorgos@cs.ucy.ac.cy

Abstract

Real-time 3D pose estimation is of high interest in interactive applications, virtual reality, activity recognition, but most importantly, in the growing gaming industry. In this work, we present a method that captures and reconstructs the 3D skeletal pose and motion articulation of multiple characters using a monocular RGB camera. Our method deals with this challenging, but useful, task by taking advantage of the recent development in deep learning that allows 2D pose estimation of multiple characters, and the

increasing availability of motion capture data. We fit 2D estimated poses, extracted from a single camera via OpenPose, with a 2D multi-view joint projections database that is associated with their 3D motion representations. We then retrieve the 3D body pose of the tracked character, ensuring throughout that the reconstructed movements are natural, satisfy the model constraints, are within a feasible set, and are temporally smooth without jitters. We demonstrate the performance of our method in several examples, including human locomotion, simultaneously capturing of multiple characters, and motion reconstruction from different camera views.

Keywords: Monocular Video, Motion Reconstruction, OpenPose, 3D Pose Estimation

1 Introduction

Motion capture (mocap) is the technological process used for acquiring three-dimensional (3D) position and orientation information of a moving object. Despite recent advances in motion capture technology, that allows high quality acquisition and portrayal of movements, common capture systems are cost-demanding, and require a setup of capturing devices that is not accessible to the general public, especially for home use. Recently, scholars turned their attention in using more affordable technologies, such as RGB or depth cameras. Capturing and reconstructing the motion of multiple characters using single, monocular cameras has a wide spectrum of applications in surveillance, human-computer interaction, activity recognition, behavioral analysis, and virtual reality. It is also of high interest in the growing gaming industry, where users require cost effective, and easy configurable systems for real-time human-machine interaction. Real-time pose estimation and 3D motion reconstruction using data only from a single RGB camera is, however, a challenging task. This is because skeletal pose estimation, using such sparse data, is a highly under-constrained problem. A human pose is usually represented and parameterized by a set of joint positions and rotations, whereas its heterogeneous, dynamic, and highly versatile nature, as well as the changes in camera viewpoint, makes motion reconstruction a difficult job.

Most papers in the literature estimate the human pose in 2D for one or multiple characters by localizing joint keypoints in pixel space [1, 2] or by extracting the shape silhouette and then retrieving the closest neighbor from a database [3, 4, 5]. More recently, and

with the advent of Deep Learning (DL), the community is moving to learning-based discriminative methods, where the effectiveness of 2D human pose estimation has greatly improved [6, 7, 8]. The 3D skeletal reconstruction, though, is a much harder problem [9, 10]. Even though there are methods that are effective at 3D pose reconstruction, they are usually not real-time implementable, and suffer from depth, and scale ambiguities. In addition, the reconstructed motion is temporally inconsistent and unstable since they treat each frame independently, and do not employ bone lengths constraints. The big challenge in this context is to learn rich features to encode depth, spatial and temporal relation of the body parts so as to ensure smooth motion reconstruction [11, 12]. While some recent methods run at high-frame (e.g., [11, 13]), they are still unsuitable for use in closely interactive characters or crowds. This is because they require a tracked bounding box for each person, and thus, they can only reconstruct the motion of a single person at a time.

The novel contribution of this paper, in principle, is that it fits 2D deep estimated poses, taken from a single, monocular camera, with the 2D multi-view joint projections of 3D motion data, to retrieve the 3D body pose of the tracked character. Our method takes advantage of the recent advances in deep and convolutional networks that allow 2D pose estimation of multiple characters, and the large (and increasing) availability of motion capture data. More particularly, our method infers the 3D human poses in real-time using only data from a single video stream. To deal with the limitations of the prior work, such as the bone length constraints violations, the simultaneously capturing of multiple characters, and the temporal consistency of the reconstructed skeletons, we generate a database with numerous 2D pro-

jections by rotating, a small angle at a time, the yaw axis of 3D skeletons. Then, we match the input 2D poses (which are extracted from a single video stream using the OpenPose network [8]) with the projections on the database, and retrieve the best 3D skeleton pose that is temporally consistent to the skeleton of the previous frames, producing natural and smooth motion.

Our approach is capable of estimating the posture of multiple characters in real-time, while the reconstructed pose is always within a natural and feasible set. The performance of our method in reconstructing 3D articulated motion from 2D poses is demonstrated in several examples, including video streams taken from different points of view, and using multiple characters in the scene.

2 Related Work

There has been a growing demand in recent years for realistic 3D animation in media and entertainment, as well as for research and training purposes. 3D motion acquisition can be roughly categorized as being achieved either using *marker-based*, or *marker-less* motion capture systems.

Marker-based systems are generally producing high-quality, realistic animations and are mainly used by the movies and entertainment industries. They use fiduciary markers which are attached near each joint to identify motion, and they provide real-time acquisition of labeled or algorithmically tracked data to reconstruct the subjects' articulated motion.

More specifically, they utilize data captured from special markers (passive and active) to triangulate the 3D position of a subject inferred from a number of high speed cameras[14, 15]. The main strengths of optical systems are their high acquisition accuracy, speed, and high sample rate capture. However, optical hardware is expensive, intrusive by nature, and lacks portability; calibration is needed for precise application of the markers, while extensive and tedious manual effort is required to recover the misapplied or (self-)occluded markers [16, 17].

More recently, a number of autonomous systems have been developed that use a variety of sensors, such as inertial measurement units (IMUs) [18, 19, 20]. Inertial mocap technology use a number of gyroscopes and accelerometers to measure rotational rates, while these rotations are translated to a skeleton model. Inertial systems do not require external cameras to capture the sensors, and thus, they are portable and functional in outdoor environments. Nevertheless, marker-based systems are costly and thus not suitable for home use, and more particularly, for interactive applications or gaming.

In general, there is a tendency to reduce the required equipment and the objects attached to the body for motion tracking and reconstruction. One way to deal with sparse data is to model the trajectories of the end effectors, and then apply kinematics models to fill in the gaps [21, 22]. The most popular way, though, is the use of **marker-less** systems (or vision systems), that are becoming more and more popular since they are easy to set-up, have low cost, and are less intrusive, meaning that subjects are not required to wear special equipment for tracking. Usually, the subject's silhouette is captured from a single or multiple angles

using a number of vision or RGB-depth cameras [23, 24]. A voxel representation of the body is extracted over time, while animation is achieved by fitting a skeleton into the 3D model, e.g., [25, 26, 27, 28, 29]. These approaches can be broadly classified into discriminative, generative and hybrid approaches. Generative methods reconstruct human pose by fitting a template model to the observed data [30, 31]. Discriminative methods infer mode-to-depth correspondences, cluster pixels to hypothesize body joint positions, fit a model and then track the skeleton [32, 33]. Hybrid methods use a combination of the aforementioned techniques to achieve higher accuracy [34]. Other works also include methods for motion capturing using an architecture of multiple color-depth sensors to better deal with occlusions or unobserved view angles [35, 36].

Skeletal pose estimation from a single camera is a much harder problem; the problem has been tackled by localizing the joint keypoint positions in pixel space [1, 2], or by extracting the shape silhouette and then retrieving the closest neighbor from a database [3, 4, 5]. For a recent overview of 3D pose estimations, refer to [37].

More recently, deep learning has brought revolutionary advances in computer vision and graphics, including efficient methods for pose estimation. Learning-based discriminative methods, are quite popular for real-time 2D pose estimation of single or multiple characters [6, 7, 38, 8]. More recently, a number of methods tackled a much harder problem, that of 3D skeletal estimation from single color images or videos [9, 10, 39, 40, 13, 41]. A common limitation of the discriminative methods in 3D pose reconstruction, though, is that they typically run off-line, and since they do not take into consideration the temporal

consistency of motion (they reconstruct the 3D joint positions on per image), the generated motion is oscillating. Moreover, they are restricted to the viewpoints learned from the training data [42], while they suffer from depth, and scale ambiguities. The problem of different camera-views can be dealt by training the network to predict the same pose using images from multiple views [43, 44]. Another major limitation is that the reconstructed 3D pose is not assigned on a character model (3D joint positions are estimated independently), to enforce kinematic constraints, resulting in temporal bone length violations. The methods in [11] (V-Nect), and [12] animate a modelled character, work in real-time, and produce smooth animations in terms of their temporal coherence. However, since they require each character to be tracked by a bounding box, they only reconstruct single-person skeletons at a time, making them unsuitable for closely interacting characters. More recently, an enormous effort has been devoted to deep and convolutional methods that map all human pixels of an RGB image to 3D surface of the human body [45, 46, 47, 48].

Our method overcomes most of the prior work limitations, including the 3D capturing of multiple characters, the skeletal model constraints, and the production of smooth animation for the articulated character.

3 Method Overview

Our approach for motion reconstruction can be decomposed into two main parts: (a) the preprocessing step, whereas a 2D pose database is defined by motion capture data projec-

tions, and its entries are associated with 3D skeletons, and (b) the run-time step, where 2D joint positions extracted from a video are matched to the 2D pose projections database in order to recover the 3D skeleton and reconstruct the motion.

More specifically, as a first step, we retarget motion data taken from an online motion capture database [49] into a universal skeleton format, and then compute the 2D projections of the 3D skeleton from many different views. The 2D projections are associated with their 3D skeletons and their viewing information, and stored in a database. The 2D projections are thereafter grouped into mutually exclusive clusters, and for each cluster, we allocate a representative pose. Then, in the second stage, for an input frame taken from a video stream, we extract in real-time its 2D pose using the OpenPose network [8]. The 2D pose is then rescaled so as to be consistent with the 2D projections stored in the database. We select the k-best matches from the database to the input 2D pose, and retrieve their 3D skeletons. To achieve smooth reconstruction of the articulated motion, we select the 3D skeleton from the k-matched projections that is temporally consistent to the previous frame.

4 Motion Database

The first step of our method, in a preprocessing time, is to define a pose database that will be used to retrieve the 3D pose estimates. We used a small dataset $\mathfrak{D}$ of mocap data, in the biovision hierarchy format (.bvh), taken from the Carnegie Mellon University (CMU) mocap library [49] (see Figure 1). The human poses in these CMU 3D data are represented

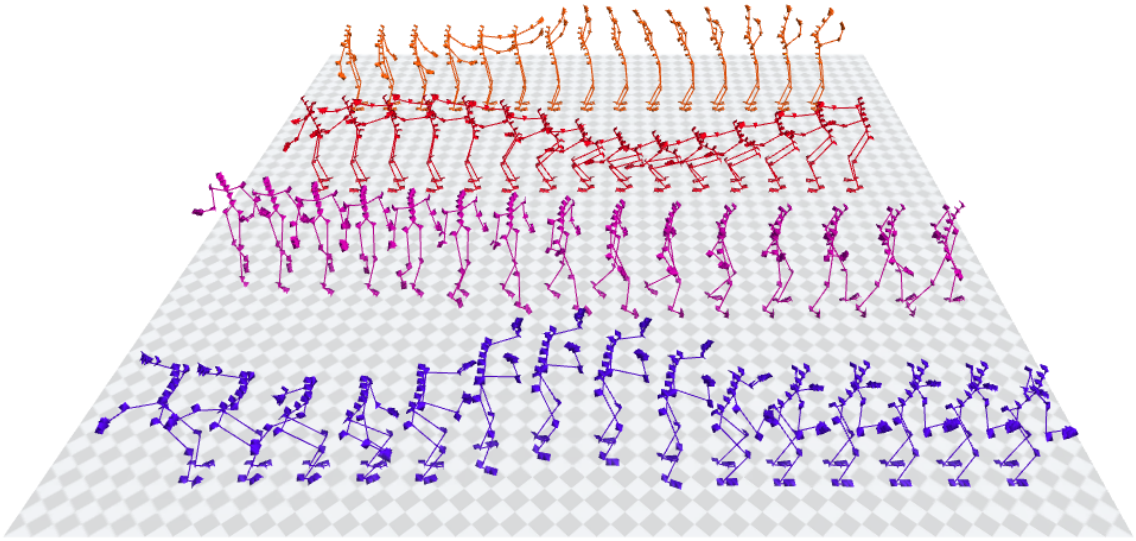

Figure 1: A number of skeleton sequences used in our dataset $\mathfrak{D}$, taken from the CMU motion capture library.

by $n = 30$ joint positions and rotations. In our work we use 2D poses that are estimated from an input monocular video using OpenPose [8] (see Section 5 for more details), that are represented by $m = 14$ joint locations (see Figure 1). Thus, in order to have a uniform and comparable skeleton, we retarget the CMU bvh data to a 3D skeleton that its projection in T-pose matches the 2D skeleton (in T-pose) returned by OpenPose (see Figure 2 for the new skeleton); the 2D pose projections are then scaled so as their bounding box remains constant over time. This step is crucial since we will compare pixelwise the joint locations of the 2D skeletons joint-by-joint.

In order to make our method invariant to the camera view, and robust against fine-grained

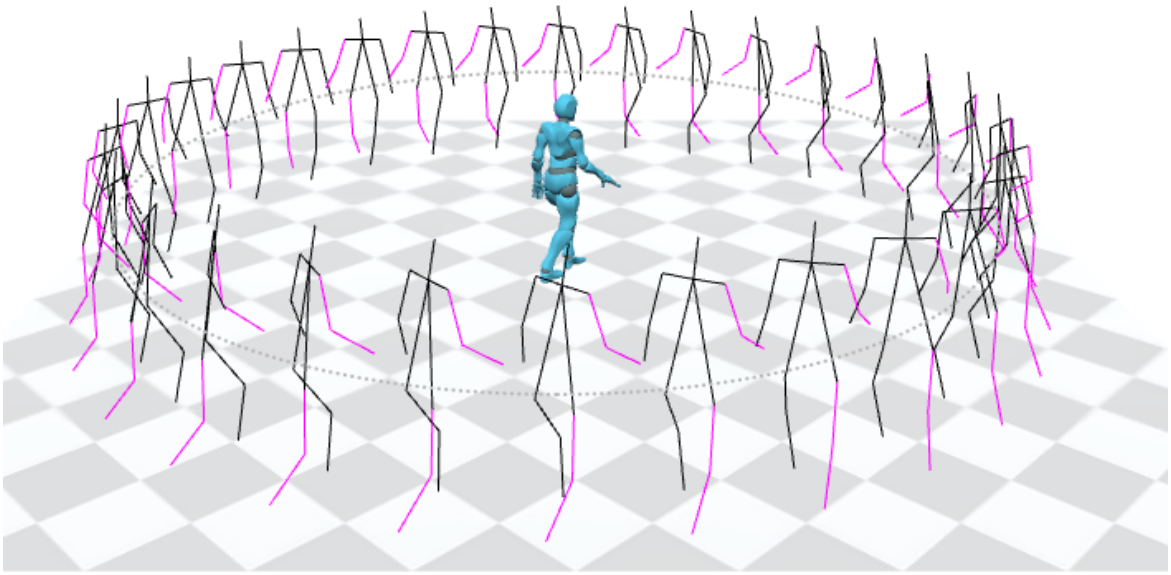

Figure 2: Multi-view skeleton projection. For each frame, we rotate the 3D skeleton by an angle θ at a time on the yaw axis, and create 2D projections.

pose variations, for each frame we compute the 2D projections of the 3D skeleton, rotated by an angle θ at a time ($\theta = 12^o$) on the yaw axis, resulting in total to 30 projections per frame. The main idea is to assign for each 3D pose multiple 2D projection representation from many different camera viewpoints, and is illustrated in Figure 2. Each 2D projection is then associated with its 3D skeleton (pointed to its frame on the bvh animation), by indexing the corresponding frame on the animation, and its rotation angle. Note that, in this work, we place the camera at 1.5 meters above earth plane (common height for video recording), and similarly, we project the 3D skeletons assuming that the camera is at the same height.

To allow fast skeleton indexing and retrieval, we position the 2D pose projections into d-dimensional space using Multi-Dimensional Scaling (MDS) [50]; we tried different di-

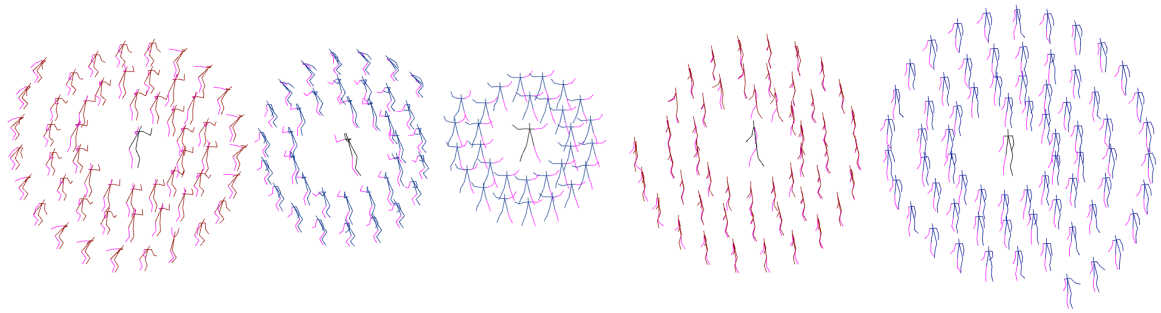

Figure 3: Five selected clusters with their representatives. The projected poses with the smallest distance to the centroid are placed nearest to the center, and as we move further from the center, the poses are less correlated to the centroid.

mensionalities, $\mathbb{R}^d$, and the quality of embedding seems to be equally well for $2 \leq d \leq 5$. The distance metric used to compute the distance between the 2D skeletal projections is defined as:

$$dist_{ij} = \sum_{k=1}^{m} d\left(\mathbf{p}_k, \mathbf{q}_k\right), \tag{1}$$

where m is the number of joints, and $\mathbf{p}_k$, $\mathbf{q}_k \in \mathbb{R}^2$ are the joint positions of the two skeletons i and j. The term $d\left(\mathbf{p}_k, \mathbf{q}_k\right)$ represents the Euclidean norm between the two joint positions.

2D projection representations in $\mathbb{R}^d$ are then grouped into mutually exclusive clusters. We use the k-means clustering algorithm to create clusters that are separated by similar characteristics. For each cluster, we also define a representative pose that is the one closest to the centroid of the cluster. Figure 3 portrays the 2D poses of five selected clusters and their representative skeletons. As can be seen, our database is rich enough and each pose has a lot of repetitions with a big variation of different poses.

5 Motion Reconstruction

For an input video stream, we use OpenPose to estimate, in real-time, the 2D pose of the characters. OpenPose is a bottom-up approach that uses Part Affinity Fields, a set of 2D vector fields that encode the location and orientation of limbs over the image domain, offering robustness to early commitment. It offers state-of-the-art accuracy, it returns the 2D key points of the joints for multiple articulated characters at the same time, but most importantly, it decouples runtime complexity from the number of people in the image, achieving low computational cost.

In this project, we assume that the character's skeleton is fully visible at the camera. Since the 2D joint locations are inferred from a video, the size of the estimated pose is depended on the size of the tracked character (e.g., adult or child), and its depth (e.g., how far or near is to the camera). To deal with this size ambiguity, and be consistent with the 2D projections stored in the database, we introduce a scaling step. For each OpenPose 2D pose, we export its bounding box and rescale it so as the height of the bounding box remains constant over time. Figure 4 shows a sequence of 2D poses of a walking character moving closer to the camera (the size of its pose increases over time), as inferred by OpenPose, and its rescaled pose that its size remains constant over time.

Having the rescaled 2D pose from OpenPose, we then need to search and retrieve the nearest pose to the entry from the projections database, and associate it with its corresponding 3D skeleton. However, one of the main challenges to deal with is that multiple 3D poses

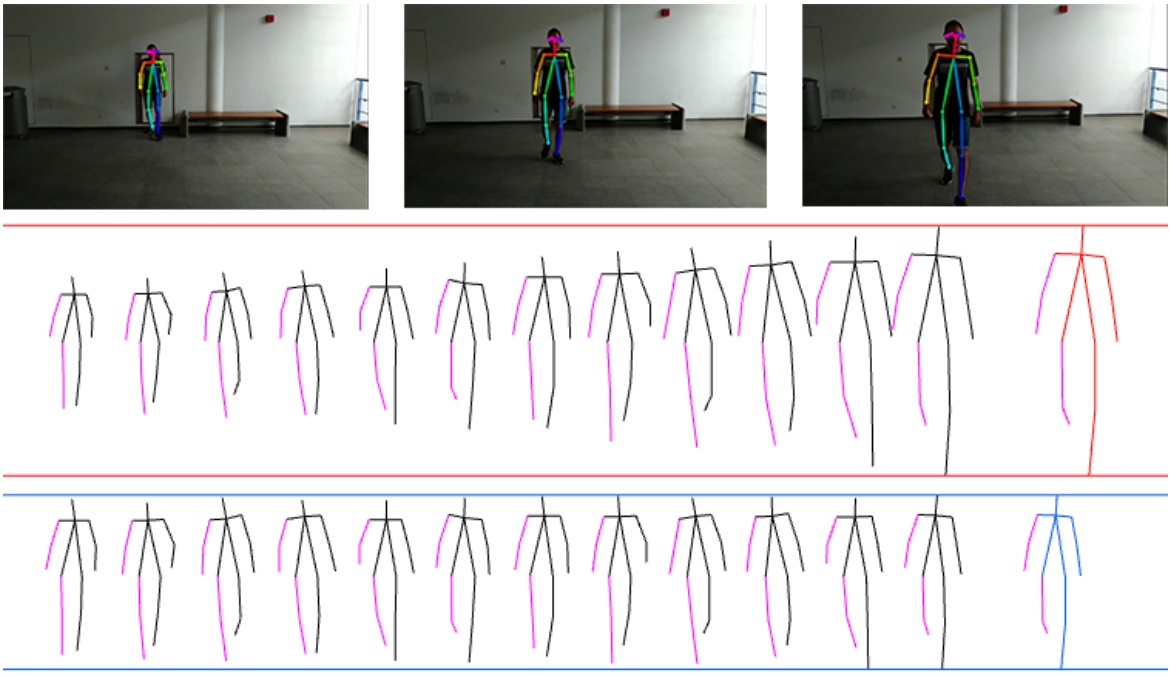

Figure 4: Scaling the OpenPose 2D pose estimation to ensure comparison consistency.

may correspond to the same 2D pose after projection. This introduces severe ambiguities in 3D pose estimation. Moreover, applying per-frame pose estimation on a motion sequence does not ensure temporal consistency of motion, and small pose inaccuracies lead to temporal jitter and motion oscillations. We dealt with these challenges in two steps. First, we compare the entry pose with the cluster representations, and then select the $i = 5$ closest clusters with the smallest Euclidean distance (see equation 1) between the input pose and their representative. Note that, searching on a single cluster is not a good idea since there is no guarantee that the closest representative contains the closest projection for our entry. For each of those i clusters, we retrieve the $j = 10$ nearest poses projections.Then, by taking advantage of the property that motion is locally linear, we retrieve the 3D skeleton that is

temporally more consistent to the reconstructed poses of the previous frames. More particularly, for each of the $l = i \times j$ selected nearest pose projections, we extract their associated 3D skeleton (their temporal location is pointed to the .bvh animation). Thereafter, we compare the 3D skeletons of their w previous frames (we empirically conclude that $w = 4$ is enough for retrieving the most temporally consistent skeleton), in the .bvh animation, with the w previously reconstructed skeletons (each of the w skeletons is weighted differently), and select the one that its w previous frames have the smallest average distance (see Figure 5 for a visual explanation). To compute the distances between the 3D skeletons, we use the Lee *et al.* [51] distance metric, that is the sum of the difference in rotation between joints. The distance between two skeletons is defined as:

$$dist_{ij}^2 = \sum_{k=1}^{m} \| \log \left(q_{j,k}^{-1} q_{i,k} \right) \|^2, \tag{2}$$

where m is the number of joints, and $q_{i,k}$, $q_{j,k} \in \mathbb{S}^3$ are the complex forms of the quaternion for the k-th joint of the two skeletons i and j, respectively. The log-norm term $\| \log \left(q_{j,k}^{-1} q_{i,k} \right) \|^2$ represents the geodesic norm in quaternion space, which yields the distance from $q_{i,k}$ to $q_{j,k}$ on $\mathbb{S}^3$. The retrieved 3D skeleton is placed at the current frame, in the animation, after it is rotated by an angle θ in the yaw-axis.

Finally, apart from retrieving the pose that is more related to the previous frames, in order to remove unwanted oscillation, we additionally smooth the reconstructed 3D motion using real-time filtering, e.g., the Savitzky-Golay [52] or the 1 € filter [53]. The filtering is applied on the joint rotations (represented in Euler angles) of the 3D skeleton. Motion

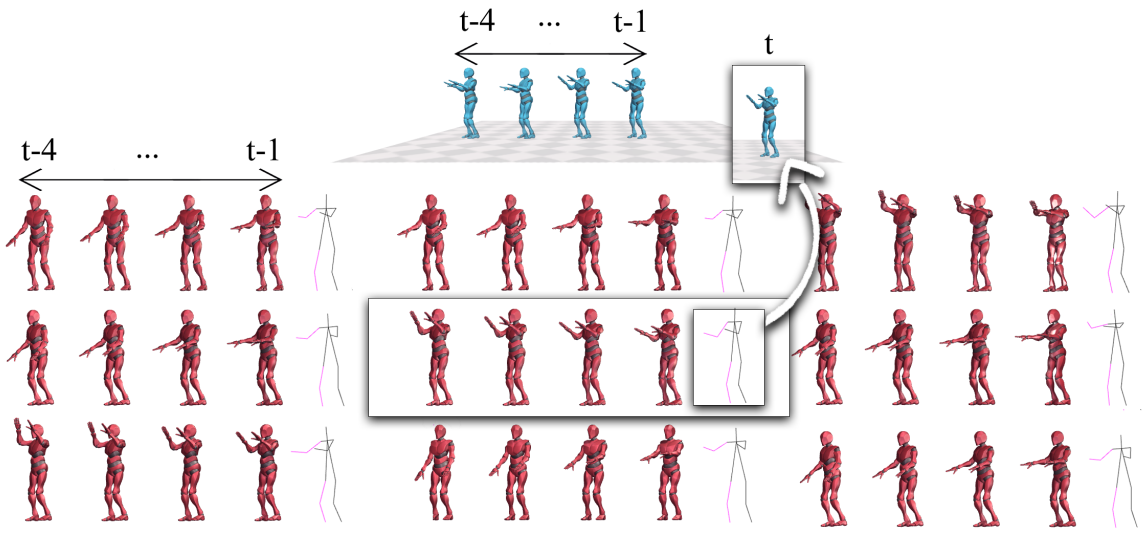

Figure 5: To ensure temporal consistency in motion, for each candidate skeleton, we compare its w previous frames in the original animation (shown in red) with the w previously reconstructed skeletons (shown in blue), and then select the candidate skeleton that its previous frames return the smallest average distance (highlighted box).

data are further edited, again in real-time, using FBBIK [54] to avoid common synthesis artifacts, such as foot sliding and floor penetration. It is important to note here that, by retrieving existing 3D skeletons we ensure that bone length are constant over time, a common limitation of prior work in 2D and 3D pose estimation. The data were transformed into .bvh format that includes absolute root position and orientation of the relative joint angles.

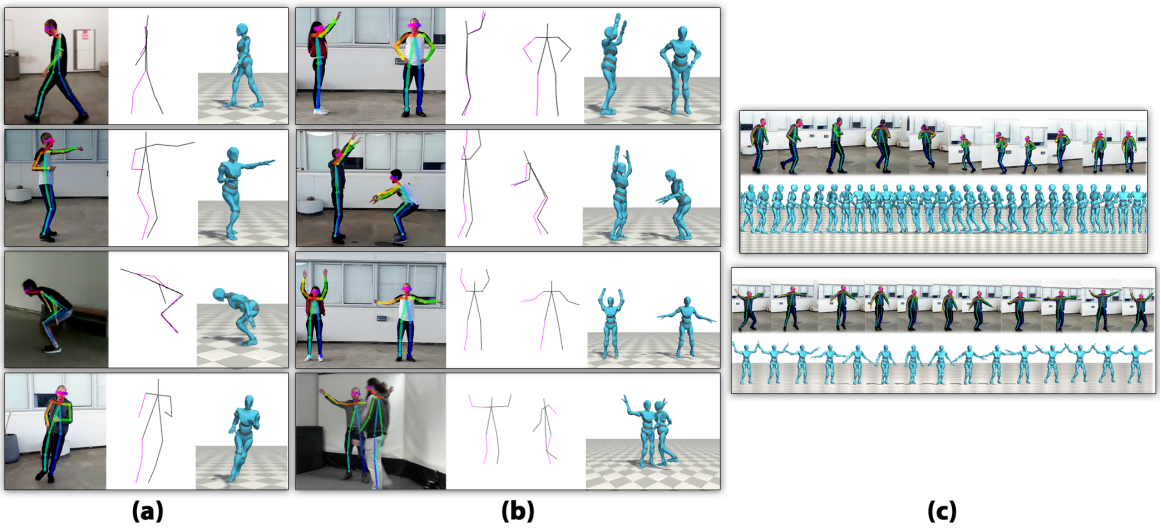

Figure 6: 3D skeleton reconstruction of various different frames for (a) single character, and (b) multiple interactive characters in the scene. The RGB video frames are shown on the left side of the figure, the 2D pose estimations, as returned by OpenPose, in the middle, and our 3D skeleton reconstructions on the right. (c) Smooth motion reconstruction in a sequence of skeletons.

6 Results and Discussion

In this section, we demonstrate the performance of our method in reconstructing 3D skeletons from RGB video sequences. We first provide the implementation details, its computational performance, and then present several experiments that illustrate the effectiveness of our work.

6.1 Implementation Details

We have implemented our system in the Unity game engine using $C\#$. All experiments were run on a six-core PC with Intel i7-6850K at 3.6GHz, 32GB RAM, and nVIDIA Titan XP GPU. We created a 2D projection database using 3D motion capture data taken from the CMU mocap library. These data were originally sampled at 120 frames per second (fps), but since human motion is locally linear, and in order to reduce the computational cost, we resample it to 24 fps without much loss of the temporal information (see [55]). In our experiments, we only used a small dataset of different actions, in total 5 minutes of motion ($72,000$ frames). Since for each frame we extract 30 projections (rotated by an angle $\theta = 12^o$ on the yaw-axis), our 2D projection database $\mathfrak{D}$ consists in total with $216,000$ pose projections. Selecting the right number of clusters for organizing $\mathfrak{D}$ is very important since it will allow fast indexing, and searching of poses, but it will also guarantee the quality of motion retrieval. We empirically concluded that our clusters are compact for roughly $K = 500$; we ended up at this number of clusters by increasing their number until the mean distance between the skeletons of each cluster and its centroid do not change more than 1%.

Our method requires approximately 12 hours to compute the distance matrix between the poses and to create the pose embedding, 14 hours for dimensionality reduction, and 15 minutes for the pose clustering (in Matlab R2018b). Although this process is time consuming, it is only required once at a pre-processing time. Once it is done, 3D motion retrieval is fast and performed in real-time (see the supplementary video for a live demonstration). There

are several factors that affect the runtime, but also the quality of the reconstructed animation, e.g., the size of the database (bulkier $\mathfrak{D}$ means larger variety of movements, thus better reconstruction quality, but it requires higher computational cost; this is mainly because of the complexity in matrix computation and MDS), the rotation angle for the projections (again, more frequent projections will increase the quality at the cost of higher computational cost).

6.2 Evaluation

We conducted several experiments to evaluate the performance of our method. Figure 6 (a) visualizes the 3D pose reconstruction on several motion examples, including walking, boxing, jumping, and running. Figure 6 (b) illustrates the corresponding results for multiple, closely interactive characters. Similarly, Figure 6 (c) shows the effectiveness of our method in tracking and capturing motion that is temporally coherent. It can be observed that the 3D skeletons of the articulated motion are smoothly reconstructed over time.

To quantitatively evaluate the performance of our method, we animated a .bvh file (that is not part of our database) using a virtual character, and then reconstruct its articulated motion (see Figure 7). We then computed the difference between the poses of the original and reconstructed characters; the average Euclidean distance between their joints is only 9.4 cm (for a character with height 175 cm). Please refer to our supplementary video for an overlay comparison of the two skeletons. We further evaluated the effectiveness of our method in pose reconstruction by recording the same action from two different viewing

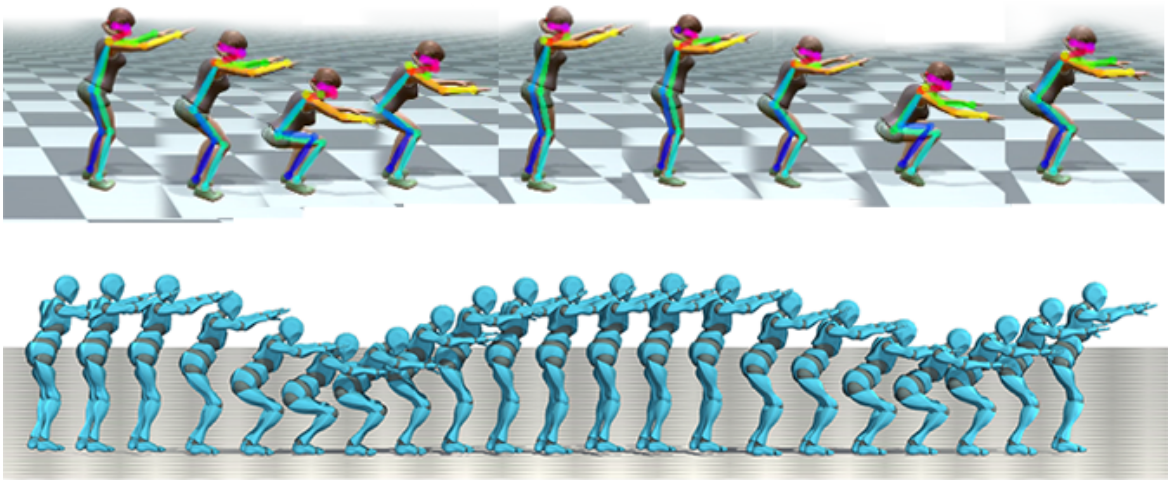

Figure 7: 3D motion reconstruction on virtually animated motion capture data.

angles, and then reconstructing their 3D motion. Again, we computed the average Euclidean distance between the two 3D skeletons per joint, that is 12.6 cm at recordings with different angle views approximately $135°$, and 6.8 cm for $75°$. Similarly, the corresponding distance for the filtered VNect reconstruction is 16.4 cm and 15.2 cm, respectively. Note that, for the comparison between the two poses, we discard the translation and rotation of the root joint so as to fairly compare the two 3D poses. Figure 8 illustrates the two 3D poses from different angles. For a side by side animated comparison of the two 3D skeletons, please refer to our accompanied video.

Results demonstrate the effectiveness of our method in reconstructing, at real-time, the articulated motion in various different actions, in video streams taken from different points of view, virtually generated videos, and using single or multiple characters in the scene. At the same time, our method ensures the smoothness and temporal consistency of motion, and

that the character's bone length constraints are not violated.

7 Conclusions, Limitations, and Future Work

We have presented a method that estimates, in real-time, the 3D pose of a human character using a single, monocular camera, and reconstructs its articulated motion. Our method is capable of tracking and reconstructing multiple 3D human postures at the same time, ensuring that the estimated 3D poses satisfy the character's bone constraints, and are always within a natural and feasible set. Common prior work limitations were efficiently dealt, such as the temporal consistency of poses, and the production of smooth and linear motion. We have evaluated the performance of our method in several examples, including a large variety of locomotion with single and multiple characters in the scene, on data taken from different points of view, artificially generated data, etc.; our results demonstrate the efficiency of the proposed approach.

Our method has some limitations. First, the accuracy of the 2D joint estimation can greatly affect our method's 3D estimation performance. This is more obvious at the hand and feet joints, which are more vulnerable to noise. There are two ways to overcome this limitation: (a) one way is to use a weighted distance metric and enforce less influence (see eq. 2) on these joints, or (b) to use an architecture of two or more cameras for a better 2D pose estimation. Moreover, in our work we assume that the character's skeleton is fully visible at the camera; however, this is not always possible due to occlusions from other

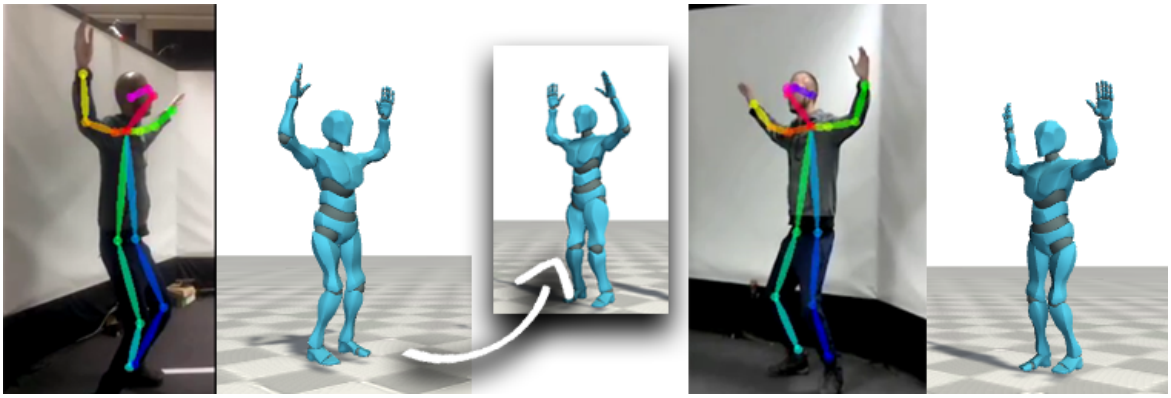

Figure 8: 3D pose reconstruction of the same action recorded from different angles. The arrow shows the same pose, but rotated, to be aligned and visually comparable with the pose from the other view.

characters or objects in the scene. Future work will see the introduction of a pose recovery method that selects the most appropriate skeleton, from the database, and matches with the sparse data.

A second limitation lies on the camera's projection angle, which must match the view angle of the camera. A possible way to make our system invariant to different camera viewpoints is to further extend the database with additional projections on different axis, different height (view), or camera parameters. 3D pose interpolations will be employed to match the 3D keypoints so as to enable 3D pose detection with different camera configurations. Nevertheless, learning the camera viewpoint to improve 3D pose estimation is currently an active and challenging topic in computer vision [56].

Another limitation of our method is that the results are highly related to the training data.

A larger variety of movements will increase the variety of reconstructed actions, but at the same time, it will increase the computational cost. Finally, since there is no calibration of the scene, our method is not trained to account for global translation and rotation. In future work, scene calibration and root translations will be added.

Acknowledgements

This work has been supported by the RESTART 2016-2020 Programmes for Technological Development and Innovation, through the Cyprus Research Promotion Foundation, with protocol number P2P/JPICH_DH/0417/0052. It has also been partly supported by the project that has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 739578 (RISE-Call: H2020-WIDESPREAD-01-2016-2017-TeamingPhase2) and the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

References

[1] L. Bourdev, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. Poselets: Body part detectors trained using 3d human pose annotations. In *Proc. of ICCV'09*, pages 1365–1372. IEEE CS, 2009.

[2] P. F. Felzenszwalb, . B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, 2010.

[3] C. Wang, Y. Wang, Z. Lin, A. L. Yuille, and W. Gao. Robust estimation of 3d human poses from a single image. In *Proc. of CVPR'14*, pages 2369–2376, DC, USA, 2014. IEEE CS.

[4] I. Akhter and M. J. Black. Pose-conditioned joint angle limits for 3d human pose reconstruction. In *Proc. of CVPR '16*, DC, USA, 2016. IEEE CS.

[5] F. D. Atrevi, D. Vivet, F. Duculty, and B. Emile. 3d human poses estimation from a single 2d silhouette. In *Proc. of VISAPP'16 - Volume 4*, pages 361–369, 2016.

[6] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *Proc. of ECCV '16*, 2016.

[7] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proc. of CVPR'16*, pages 4724–4732, DC, USA, 2016. IEEE CS.

[8] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. In *Proc. of CVPR '18*, DC, USA, 2018. IEEE CS.

[9] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it SMPL: Automatic estimation of 3d human pose and shape from a single image. In *Proc. of ECCV '17*, 2017.

[10] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis. Learning to estimate 3d human pose and shape from a single color image. In *Proc. of CVPR'18*, DC, USA, 2018. IEEE CS.

[11] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Trans. Graph.*, 36(4):44:1–44:14, 2017.

[12] J. Martinez, R. Hossain, J. Romero, and J. J. Little. A simple yet effective baseline for 3d human pose estimation. In *Proc. of ICCV'17*, pages 2659–2668, 2017.

[13] K. Wang, L. Lin, C. Jiang, C. Qian, and Pengxu W. 3d human pose machines with self-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.

[14] PhaseSpace Inc. Optical MoCap Systems: `http://www.phasespace.com`, 2019.

[15] Vicon. Vicon Motion Capture Systems: `http://www.vicon.com`, 2019.

[16] A. Aristidou and J. Lasenby. Real-time marker prediction and CoR estimation in optical motion capture. *Vis. Comput.*, 29(1):7–26, 2013.

[17] A. Aristidou, D. Cohen-Or, J. K. Hodgins, and A. Shamir. Self-similarity analysis for motion capture cleaning. *Comput. Graph. Forum*, 37(2):297–309, 2018.

[18] Xsens Technologies B.V. Motion Capture Systems: `http://www.xsens.com`, 2019.

[19] J. Tautges, A. Zinke, B. Krüger, J. Baumann, A. Weber, T. Helten, M. Müller, H.-P. Seidel, and B. Eberhardt. Motion reconstruction using sparse accelerometer data. *ACM Trans. Graph.*, 30(3):18:1–18:12, 2011.

[20] R. Slyper and J. K. Hodgins. Action capture with accelerometers. In *Proc. of SCA'08*, pages 193–199, Aire-la-Ville, Switzerland, 2008. EG Association.

[21] A. Aristidou, Y. Chrysanthou, and J. Lasenby. Extending FABRIK with model constraints. *Comput. Animat. Virtual Worlds*, 27(1):35–57, 2016.

[22] P. Carreno-Medrano, S. Gibet, and P.-F. Marteau. From expressive end-effector trajectories to expressive bodily motions. In *Proc. of CASA '16*, pages 157–163, NY, USA, 2016. ACM.

[23] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, and T. Brox. 3d human pose estimation in RGBD images for robotic task learning. In *Proc. of ICRA '18*, 2018.

[24] A. Biswas, H. Admoni, and A. Steinfeld. Fast on-board 3d torso pose recovery and forecasting. In *Proc. of ICRA'19*, 2019.

[25] G. K. M. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. In *Proc. of CVPR'03*, pages 77–84, DC, USA, 2003. IEEE CS.

[26] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. Performance capture from sparse multi-view video. *ACM Trans. Graph.*, 27(3):98:1–98:10, 2008.

[27] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. *ACM Trans. Graph.*, 27(3):97:1–97:9, 2008.

[28] J. Gall, A. Yao, and L. Van Gool. 2d action recognition serves 3d human pose estimation. In *Proc. of ECCV'10: Part III*, pages 425–438, Berlin, Heidelberg, 2010. Springer-Verlag.

[29] Y. Liu, J. Gall, C. Stoll, Q. Dai, H.-P. Seidel, and C. Theobalt. Markerless motion capture of multiple characters using multiview image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2720–2735, 2013.

[30] X. Wei, P. Zhang, and J. Chai. Accurate realtime full-body motion capture using a single depth camera. *ACM Trans. Graph.*, 31(6):188:1–188:12, 2012.

[31] M. Ye and R. Yang. Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera. In *Proc. of CVPR'14*, pages 2353–2360, DC, USA, 2014. IEEE CS.

[32] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Commun. ACM*, 56(1):116–124, 2013.

[33] T. Sharp. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *Proc. of CVPR'12*, pages 103–110, DC, USA, 2012. IEEE CS.

[34] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Proc. of ICCV'11*, pages 1092–1099. IEEE CS, 2011.

[35] D. Vlasic, P. Peers, I. Baran, P. Debevec, J. Popović, S. Rusinkiewicz, and W. Matusik. Dynamic shape capture using multi-view photometric stereo. *ACM Trans. Graph.*, 28(5):174:1–174:11, 2009.

[36] K. Berger, . Ruhl, Y. Schroeder, C. Bruemmer, A. Scholz, and M. Magnor. Markerless motion capture using multiple color-depth sensors. In P. Eisert, J. Hornegger, and K. Polthier, editors, *Proc. of Vision, Modeling, and Visualization*. EG Association, 2011.

[37] N. Sarafianos, B. Boteanu, B. Ionescu, and I. A. Kakadiaris. 3d human pose estimation: A review of the literature and analysis of covariates. *Comput. Vis. Image Underst.*, 152(C):1–20, 2016.

[38] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy. Towards accurate multi-person pose estimation in the wild. In *Proc. of CVPR'18*, DC, USA, 2017. IEEE CS.

[39] X. Zhou, M. Zhu, G. Pavlakos, S. Leonardos, K. G. Derpanis, and K. Daniilidis. MonoCap: Monocular human motion capture using a CNN coupled with a geometric prior. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018.

[40] W. Yang, W. Ouyang, X. Wang, J. S. J. Ren, H. Li, and X. Wang. 3d human pose estimation in the wild by adversarial learning. In *Proc. of CVPR'18*, pages 5255–5264, DC, USA, 2018. IEEE CS.

[41] Pavllo. D., C. Feichtenhofer, D. Grangier, and M. Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proc. of CVPR'19*, DC, USA, 2019. IEEE CS.

[42] E. Simo-Serra, A. Quattoni, C. Torras, and F. Moreno-Noguer. A joint model for 2d and 3d pose estimation from a single image. In *Proc. of CVPR'13*, pages 3634–3641, DC, USA, 2013. IEEE CS.

[43] H. Rhodin, J. Spörri, I. Katircioglu, V. Constantin, F. Meyer, E. Müller, M. Salzmann, and P. Fua. Learning monocular 3d human pose estimation from multi-view images. In *Proc. of CVPR'18*, pages 8437–8446, DC, USA, 2018. IEEE CS.

[44] H. Rhodin, M. Salzmann, and P. Fua. Unsupervised geometry-aware representation for 3d human pose estimation. In *Proc. of ECCV'18*, 2018.

[45] C. Lassner, J. Romero, M. Kiefel, F. Bogo, M. J. Black, and P. V. Gehler. Unite the people: Closing the loop between 3d and 2d human representations. In *Proc. of CVPR'17*, pages 4704–4713, DC, USA, 2017. IEEE CS.

[46] A. Kanazawa, J. Black, M, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. In *Proc. of CVPR'18*, pages 7122–7131, DC, USA, 2018. IEEE CS.

[47] W. Xu, A. Chatterjee, M. Zollhöfer, H. Rhodin, D. Mehta, H.-P. Seidel, and C. Theobalt. Monoperfcap: Human performance capture from monocular video. *ACM Trans. Graph.*, 37(2):27:1–27:15, May 2018.

[48] R. A. Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proc. of CVPR'18*, pages 7297–7306, DC, USA, 2018. IEEE CS.

[49] Carnegie Mellon University. MoCap Library: `http://mocap.cs.cmu.edu/`, 2019.

[50] G. A. F. Seber. *Multivariate Observations*. John Wiley & Sons, Inc., NJ, USA, 1984.

[51] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, 2002.

[52] S. J. Orfanidis. *Introduction to Signal Processing*. Prentice-Hall, Inc., NJ, USA, 1995.

[53] G. Casiez, N. Roussel, and D. Vogel. 1 € filter: A simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of CHI '12*, pages 2527–2530, NY, USA, 2012. ACM.

[54] Root-Motion. FINAL-IK: `http://root-motion.com/`, 2019.

[55] K. Forbes and E. Fiume. An efficient search algorithm for motion data using weighted PCA. In *Proc. of SCA '05*, pages 67–76, 2005.

[56] M. F. Ghezelghieh, R. Kasturi, and S. Sarkar. Learning camera viewpoint using CNN to improve 3d body pose estimation. In *Proc. of 3DV '16*, 2016.